Projet semestriel C++

Chloé Bensoussan

Universite Nice Sophia-Antipolis
Master 1 Informatique

1 Introduction

Ce projet consiste à réaliser un jeu de Tower Defense. C'est un projet individuel, permettant d'acquérir le langage C++, débuté cette année.

Dans ce rapport je vais aborter la structure des différentes classes puis leurs implémentations qui réalisent ce jeu.

2 Les Modèles

Je vais vous présenter les quatres modèles utilisés dans ce jeu.

2.1 Vaisseaux

```
class Vaisseau {
    float x, y;
    float frequence, vitesse, puissance;
    int distance;
    int vie;
    int prix;

    float red, green, blue;
    float redInit, greenInit, blueInit;

    std::deque<Missile> missiles;
};
```

Les coordonnées `x, y` représente les coordonnées du milieu du vaisseau.

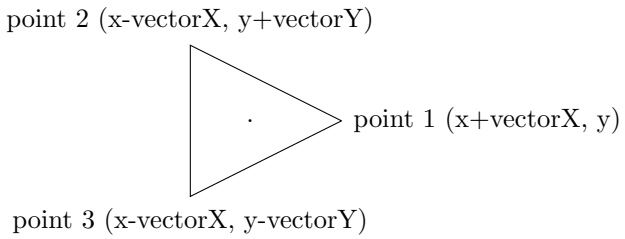
Sa `distance` représente le nombre de cases que peut atteindre un missile.

Les variables `fréquence`, `vitesse` et `puissance` sont les caractéristiques des missiles.

Chaque vaisseau possède une queue de Missiles qui se génère automatiquement et en continue (il n'y aura jamais de vaisseaux sans missiles).

Sa vie est initialisé à 15 pour tous types de vaisseaux.

Le vaisseau est représenté par un triangle en fonction de son centre et de la taille d'une case :



La destruction des vaisseaux s'effectue lorsque sa vie atteint 0.

2.2 Missiles

```
class Missile {
    float x, y, vector;
    float vposX;
    const float r = 0.0f, g = 1.0f, b = 0.0f;
};
```

Un missile est dessiné par un trait dont les coordonnées `x, y` représentent le point le plus à gauche et le `vector` sa longueur.

Un nouvel missile est créé tous les $(\frac{1}{fréquence}) \times 200$ appels de tick.
La vitesse des missiles se calcule par $VposX = \frac{vitesse}{5}$.
Sa longueur varie avec la puissance du vaisseau : $vector = puissance \times 0.03$

Le missile se détruit lorsque son coordonnée x se retrouve hors de la fenêtre ou bien lors d'une collision avec un astéroide.

Les missiles ne peuvent détruire que les astéroides que par son point le plus à droite. Il n'y a aucun impact sur les autres vaisseaux placés devant eux.

2.3 Vague

```
class Vague {
    private :
    int nombre;
    const float intervalle = 0.15;

    static int totalVagues;

    public :
    static std::deque<Asteroide> asteroides;
};
```

Lorsqu'une nouvelle vague se crée, son `nombre` d'astéroïdes augmente d'un. L'`intervalle` représente l'intervalle de temps d'apparition de chaque astéroide dans le jeu qui sont stockés dans une queue.

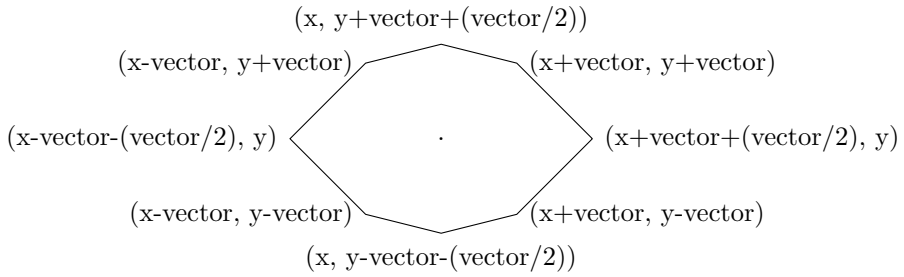
Les astéroides seront placés sur différentes lignes choisies aléatoirement.

2.4 Astéroïdes

```
class Asteroide {
    float centreX, centreY;
    float vector;
    float red, green, blue;
    float vitesse;

    int vie;
    std::vector<float> x;
    std::vector<float> y;
};
```

Un astéroïde est représenté par un polygone en fonction des coordonnées du centre et de son *vector* :



3 Classe Jeu

Dans mon programme il existe une classe `Jeu` qui possède tous les attributs et méthodes `static` du jeu, permettant d'y accéder depuis les autres classes. C'est pourquoi il n'y a aucun objet de ce type créé.

```
class Jeu {
    private:
    static const int nb_lignes = 5;
    static int vie;
    static int totalVaisseaux;
    static int tirelire;
    static std::string message;
    static bool pause;

    public :
    static std::vector<Vaisseau> typesVaisseaux;
    static std::vector<Asteroide> typesAsteroides;
```

```
    static Vaisseau choix;
    static Vague vague;
    static bool finJeu;
};
```

Le jeu est initialisé à 20 vies et 50$. Il possède 4 sortes de vaisseaux que nous pouvons choisir puis 5 sortes d'astéroïdes plus ou moins puissants, générés aléatoirement.

3.1 Collisions

Lors de la programmation de ce jeu, deux types de collision deviennent importantes : la collision entre missiles-astéroïdes puis celle entre astéroïdes-vaisseaux.

Lorsqu'il y a un missile qui touche un astéroide, l'astéroide perd de la vie (la puissance du vaisseau). Si l'astéroïde n'a plus de vie, elle disparait et nous gagnons de l'argent.

Si la puissance du missile est supérieur à $2.5 \times vie$ de l'astéroide, le missile "reste en vie" puis peut en tuer d'autres par la suite. Sinon le missile est détruit en même temps que l'astéroide.

La collision entre un vaisseau et un astéroïde se base sur le même principe : lorsqu'un astéroïde touche un vaisseau, l'astéroide perd de sa vie, mais il en fait aussi perdre à celui du vaisseau. Lorsque le vaisseau atteint 0 vie, il disparait. Dans ce cas de collision, il n'y a aucun argent gagné.

4 Conclusion

Le sujet étant assez simple (dans le sens où il n'y avait pas beaucoup de règles à respecter), nous avons pu programmer librement notre imagination pour réaliser ce jeu.

Je me suis bien amusée à coder ce jeu, en essayant différentes applications de tower defense déjà existantes, en cherchant des options pratiques et facile à implémenter puis en écoutant les avis des camarades.

La structure des classes n'étant pas forcément l'idéale mais j'ai pu apprendre beaucoup de choses sur ce langage.

En espérant que ce jeu vous plaise.
Merci Beaucoup !